Barcelona Notes on Computational Mechanics PATC Cource 2016

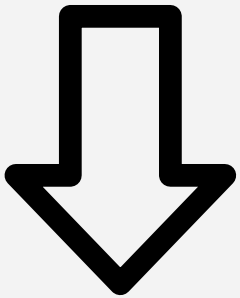
Mariano Vázquez

Barcelona Supercomputing Center
Centro Nacional de Supercomputación
España

Outline

What is behind a parallel simulation code?

What is the use of it?



Industrial Applications

The real world

Barcelona Supercomputing Center

Outline

Background

Motivation

The Physical System and its Mathematical Description

Discretization: Divide and Conquer

Discretization: Algorithms and Codes

Applications

Barcelona Supercomputing Center

Background

Barcelona Supercomputing Center

Motivation

Barcelona Supercomputing Center

Introduction: Motivation

Physical Understanding:

We deal with Physical systems

What is behind a simulation code?

Mathematical Models:

Governed by Differential Equations...

... Numerically solved

Computer Science:

... and translated in a Computational Model

Barcelona SuperComputing Center

Computational Mechanics:
A definition as a discipline on its own

Oden, Belytschko, Babuska and Hughes (2003):

> **Theoretical and applied mechanics (TAM)** is the branch of applied science concerned with the study of **mechanical phenomena**: the behavior of fluids, solids, and complex materials under the actions of forces.[...]
>
> **Computational mechanics (CM)** is that sub-discipline of TAM concerned with the use of **computational methods and devices** to study events governed by the principles of mechanics.

High Performance Computational Mechanics:

A CM sub-discipline

Efficient use of **HPC resources**

Barcelona Supercomputing Center

Computational Science or Scientific Computing:
A definition as a discipline on its own (again)

Wikipedia:

> Computational Science is concerned with constructing **mathematical models** and using **computers** to analyze and solve scientific problems.
>
> In practical use, it is typically the application of **computer simulation** and other forms of computation from numerical analysis and theoretical **computer science** to problems in **various scientific disciplines**.

Shortly, **Computational Physics or Computational Engineering**

High Performance Computational Science /Scientific Computing

Barcelona Supercomputing Center

Introduction: Motivation

Theoretical Physics

Physical model definition
Classical, Quantum, Relativistic...
Fluid, solids, electromagnetism, gravitation, rigid body...

Experimental Physics

Experiments and observation
Wind tunnels, particle accelerators, meteorology, astrophysics...

Computational Physics

Numerical methods and programming

Application areas:
Engineering, Aeronautics, Meteorology, Biology, Astrophysics...

Barcelona Supercomputing Center

Introduction: Motivation

Mature

Deep mathematical basis, reliable (not just finite differences)

Very flexible and powerful programming tools (languages, compilers, ...)

The only way to attack some problems (a lot of problems indeed...)

It allows to verify and improve the theory and design new experiments

Large (and growing) computers available

Computational Physics

Barcelona Supercomputing Center

The Physical System and its Mathematical Description

Barcelona Supercomputing Center

Revisiting the definition of CM...

Governed by forces, or more generally by conservation principles.

Usually modeled by Partial or Ordinary Differential Equations (PDE - ODE) but maybe...

... many of them and coupled (i.e. combustion, species transport...)

... multiphysics (Fluid-structure Interaction -FSI-, multiscale modelling...)

... non-local phenomena ("all against all" connectivities, infinite speed...)

... design variables exist (optimization problem)

Usually highly non-linear

Usually many of these features appears at the same time... i.e. **large problems**.

Barcelona Supercomputing Center

All these features strongly condition the method you use:

Either if you use your own code

or

if you use another one's code (commercial or open source)

The consequence: know deeply the Physics!

Barcelona Supercomputing Center

All these features strongly condition the method you use:

- Physical modeling
  - ☑ Transient vs. Stationary: *time step? what's that? how do I set it?*
  - ☑ Incompressible vs. Compressible: *Mach number in water? Shock waves?*
  - ☑ RANS vs. LES turbulence models: *what scales do I need to solve? application ranges?*
  - ☑ Material design: *what material do I have to use?*
  - ☑ Fluid-Structure Interaction: *is fluid deforming the structure? Transient?*
  - ☑ Boltzman, SPH, ...: *what is this? what are the limitations? where are they coming from?*

Barcelona Supercomputing Center

All these features strongly condition the method you use:

Physical modeling

- ☑ Transient vs. Stationary: *robustness? convergence?*
- ☑ Incompressible vs. Compressible: *convergence? discontinuities?*
- ☑ RANS vs. LES turbulence models: *how do I couple the problem? convergence?*
- ☑ Material design: *what solution scheme? do I have the tangent matrix?*
- ☑ Fluid-Structure Interaction: *how do I couple the problem? added mass?*
- ☑ Boltzman, SPH, ...: *how do they behave with respect to other methods?*

Mathematical modeling

- ☑ ... same as above, but with mathematical perspective
- ☑ Optimization problem? Adjoint vs. Genetic Algorithms

Barcelona Supercomputing Center

The Physical System

All these features strongly condition the method you use:

Physical modeling

- ☑ Transient vs. Stationary
- ☑ Incompressible vs. Compressible
- ☑ RANS vs. LES turbulence models
- ☑ Material design
- ☑ Fluid-Structure Interaction
- ☑ Boltzman, SPH, ...

Mathematical modeling

- ☑ ... same as above, but with mathematical perspective
- ☑ Optimization problem? Adjoint vs. Genetic Algorithms

Discretization

- ☑ FEM, FV, FD, Spectral method, Lattice Boltzmann, Clustering in SPH...
- ☑ XFEM
- ☑ Stabilization problems
- ☑ Time Integration: Explicit - Implicit

Barcelona Supercomputing Center

All these features strongly condition the method you use:

- Solution Algorithm
  - ☑ Time advance
  - ☑ Space solver: Direct vs. Iterative
  - ☑ Non-linear solver: Jacobi, Newton...
  - ☑ Preconditioners
  - ☑ Coupling strategies

Barcelona Supercomputing Center

All these features strongly condition the method you use:

Solution Algorithm

- ☑ Time advance
- ☑ Space solver: Direct vs. Iterative
- ☑ Non-linear solver: Jacobi, Newton...
- ☑ Preconditioners
- ☑ Coupling strategies

Implementation

- ☑ FEM, FV, FD, Spectral method
- ☑ Lattice Boltzmann, Clustering in SPH...
- ☑ Parallel vs. sequential
- ☑ OpenMP vs. MPI

Barcelona Supercomputing Center

All these features strongly condition the method you use:

Solution Algorithm

- ☑ Time advance
- ☑ Space solver: Direct vs. Iterative
- ☑ Non-linear solver: Jacobi, Newton...
- ☑ Preconditioners
- ☑ Coupling strategies

Implementation

- ☑ FEM, FV, FD, Spectral method
- ☑ Lattice Boltzmann, Clustering in SPH...
- ☑ Parallel vs. sequential
- ☑ OpenMP vs. MPI

Validation

- ☑ How can I be sure of what I am simulating?
- ☑ Study the application ranges

Barcelona Supercomputing Center

We will focus in systems governed by **Conservation Laws**

Conservation Laws come from basic Physical principles

Examples: Mass, Energy, Linear or Angular Momentum, Spin, People...

Based in:

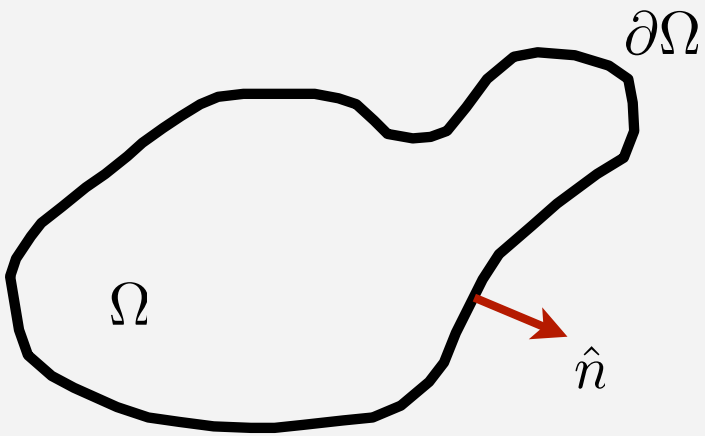
$q^{\beta}(\bar{x}, t)$	A number of certain evolving quantities
$\Omega \subset R^3$	A closed domain
$\partial\Omega$	and its boundary
$0 < t < T$	The time interval

Barcelona Supercomputing Center

Let us suppose that the domain is not changing form with time.

Then

$$\frac{\partial}{\partial t}\int_{\Omega} q^{\alpha}(\bar{x},t)dV + \int_{\partial\Omega} F_i^{\alpha}(q^{\beta}(\bar{x},t))n_i dS = 0$$



The Integral Form (IF)

$F_i^{\alpha}(q^{\beta}(\bar{x},t))$ The quantities flux

- Latin index: cartesian dimensions
- Greek index: variables that define the system

Barcelona Supercomputing Center

$$\frac{\partial}{\partial t}\int_{\Omega} q^{\alpha}(\bar{x},t)dV + \int_{\partial\Omega} F_i^{\alpha}(q^{\beta}(\bar{x},t))n_i dS = 0 \qquad \forall\alpha$$

This equation is **the fundamental mathematical interpretation of a conservation principle**

What do we mean by a **conservation principle** ?
Example: persons in this room, money in your pocket...

The principle is basic... what are the conserved quantities is not so basic. What are the fluxes is not so basic neither.
Examples: mass, mechanical energy (1st. law of thermodynamics), linear momentum (2nd. law of Newton)

Indeterminacies: fluxes definition, quantities meaning, boundary and initial values, material properties...

The conservation law gives the idea of how extensive quantities behave

Sources and drains can be included

The system can be divided in parts (volume/surface) and quantities transferred back and forth

Barcelona Supercomputing Center

$$\frac{\partial}{\partial t}\int_{\Omega} q^{\alpha}(\bar{x},t)dV + \int_{\partial\Omega} F_i^{\alpha}(q^{\beta}(\bar{x},t))n_i dS = 0 \qquad \forall\alpha$$

If the domain is changing in time, no problem: *Reynolds transport theorem*

The domain could be extended to the infinite or reduced to infinitesimal

Applying the *Gauss theorem, and requiring some continuity properties on the fluxes*

$$\frac{\partial}{\partial t}\int_{\Omega}\left(q^{\alpha}(\bar{x},t) - \frac{\partial F_i^{\alpha}(q^{\beta}(\bar{x},t)}{\partial x_i}\right)dV = 0 \qquad \forall\alpha$$

Therefore

$$\frac{\partial q^{\alpha}}{\partial t} + \frac{\partial F_i^{\alpha}(q)}{\partial x_i} = 0 \qquad \forall\alpha$$

Barcelona Supercomputing Center

The target:

Systems governed by PDEs coming from conservation principles

Forces, Energy, Mass... + Boundary and initial conditions

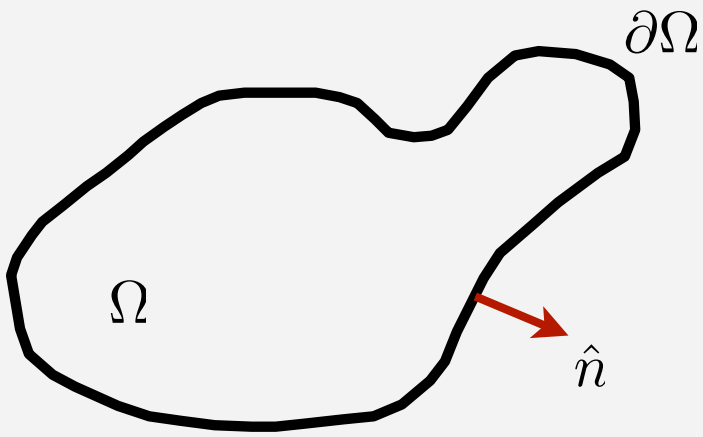
$$\frac{\partial q^{\alpha}}{\partial t} - \frac{\partial F_i^{\alpha}(q^{\beta})}{\partial x_i} = S$$

The Differential Conservative Form (DCF)

$q^{\alpha}(x_i, t)$ ← Fields of unknowns

$F_i^{\alpha}(q^{\beta}, x_i, t)$ ← Fluxes

$S(x_i, t)$ ← Sources



Barcelona Supercomputing Center

The target:

Systems governed by PDEs coming from conservation principles

Forces, Energy, Mass... + Boundary and initial conditions

$$\frac{\partial q^{\alpha}}{\partial t} - \frac{\partial F_i^{\alpha}(q^{\beta})}{\partial x_i} = S$$

The Differential Conservative Form (DCF)

Complex Physics, even for the simplest system

Complex geometries

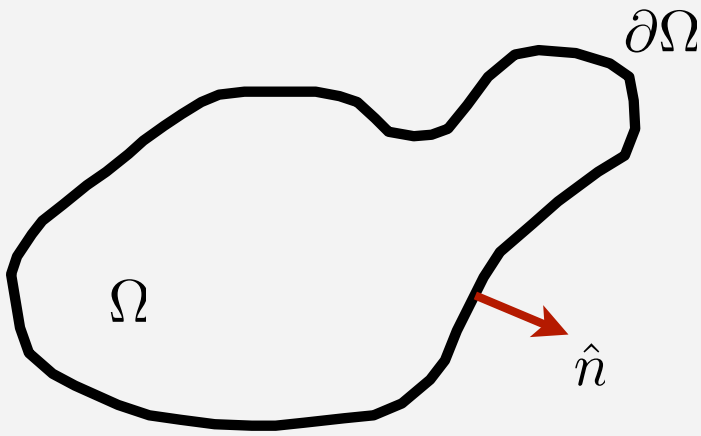
Complex numerical issues

Fine resolution in time and space

Simplifications at very high level

Coupling

No analytical solutions!



Barcelona Supercomputing Center

Discretizing the System

What is a simulation code?

"Divide and conquer"

Heavily reducing the dimensionality of the unknowns

Going from time-space continuum to a discrete subset

How to choose this **discrete subset** lies in the core of Computational Mechanics

For instance:

- ☑ Replace ∂ by Δ in the original continuum equations
- ☑ Discretize the domain in small cells and solve individual conservation problems
- ☑ Discretize the space to find an approximate solution

Barcelona Supercomputing Center

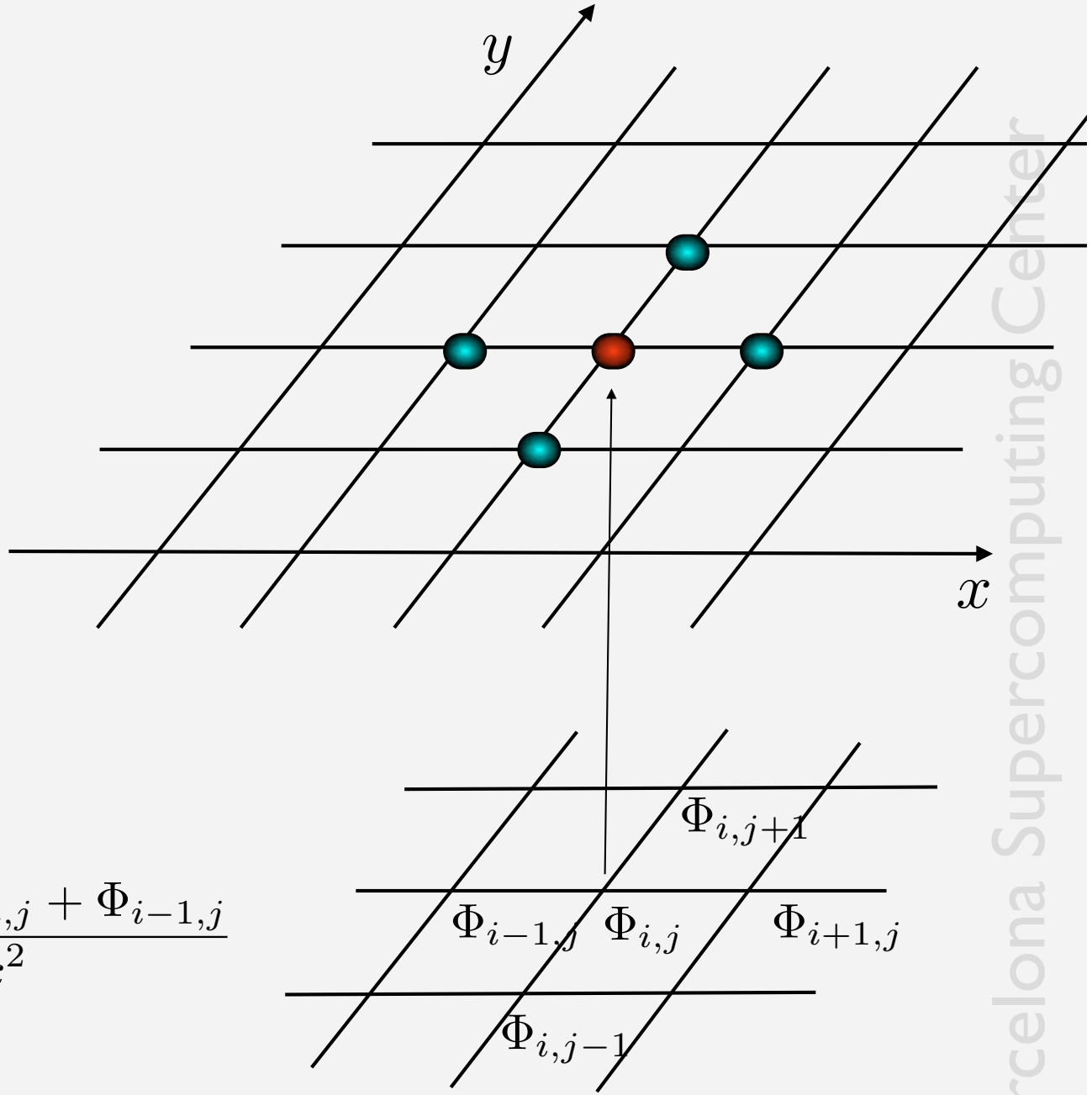
Discretization:

Divide and Conquer

Barcelona Supercomputing Center

First strategy:
Finite Differences

$$\frac{\partial q^\alpha}{\partial t} - \frac{\partial F_i^\alpha(q^\beta)}{\partial x_i} = S$$



Suppose convective and diffusive fluxes:

$$\frac{\Delta \Phi_{i,j}}{\Delta x} = \frac{\Phi_{i+1,j} - \Phi_{i,j}}{\Delta x}$$

$$\frac{\Delta}{\Delta x}\left(\frac{\Delta \Phi_{i,j}}{\Delta x}\right) = \frac{\Phi_{i+1,j} - 2\Phi_{i,j} + \Phi_{i-1,j}}{\Delta x^2}$$

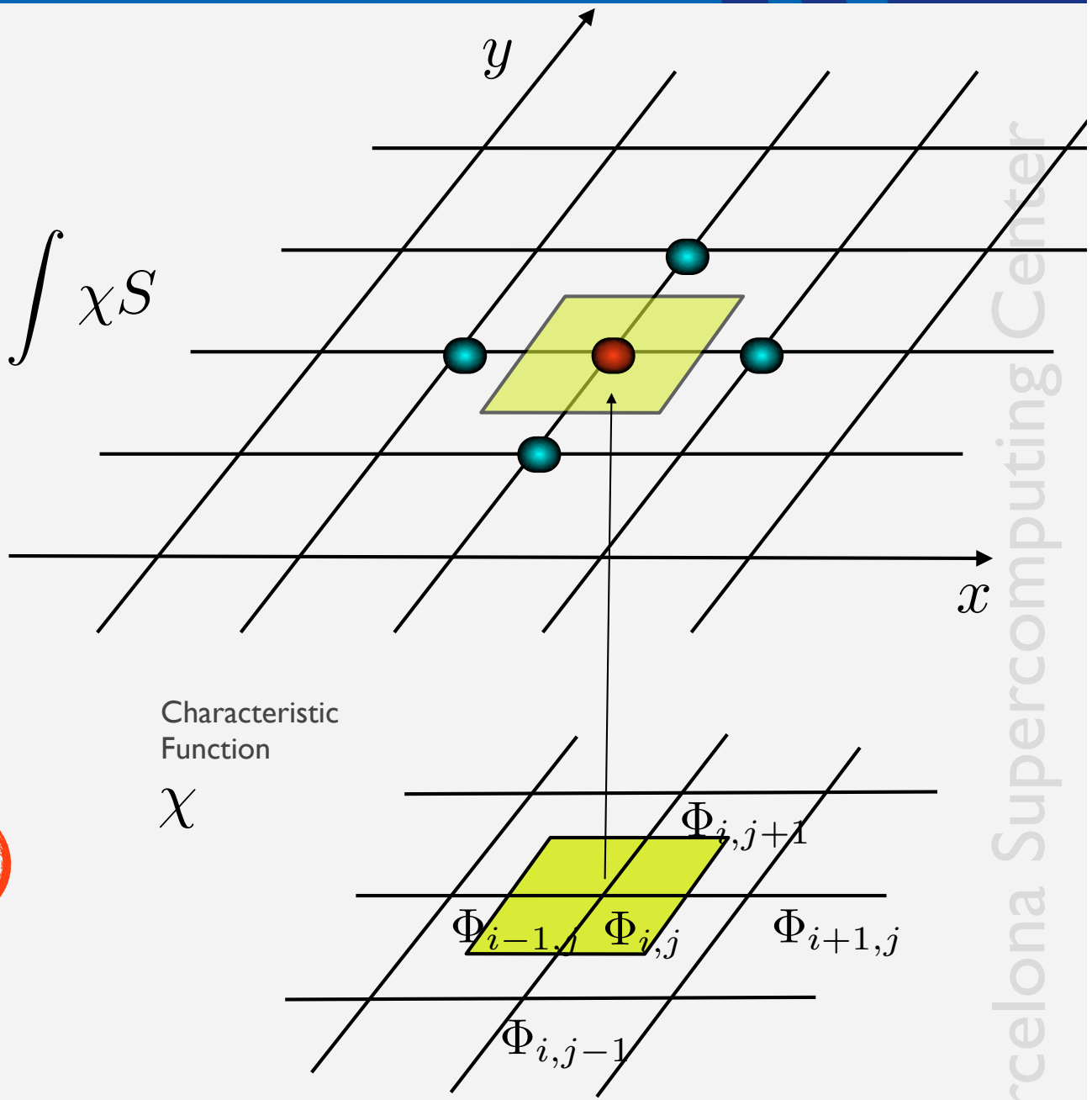
Barcelona Supercomputing Center

Second strategy:
Finite Volumes

$$\int \chi \frac{\partial \Phi^\alpha}{\partial t} + \int \chi \frac{\partial F_i^\alpha}{\partial x_i} = \int \chi S$$

Divergence Gauss Theorem

$$-\oint \chi F_i^\alpha n_i$$



Barcelona Supercomputing Center

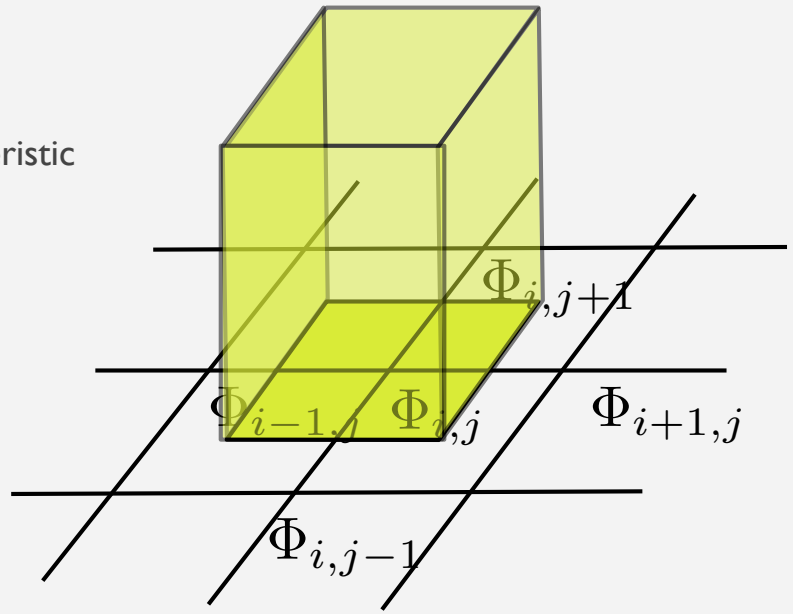
Discretizing the System

Second strategy:
Finite Volumes

$$-\oint \chi F_i^{\alpha} n_i$$

Characteristic Function

χ



The characteristic function is a "filter" that focuses the equation only in the filter's support.

When you project, the rest of the domain disappears.

Is like the Integral Form but filtered on small cells.

You only have to compute the fluxes through the limits of the cell.

These limits are discretized in faces or edges following the space discretization.

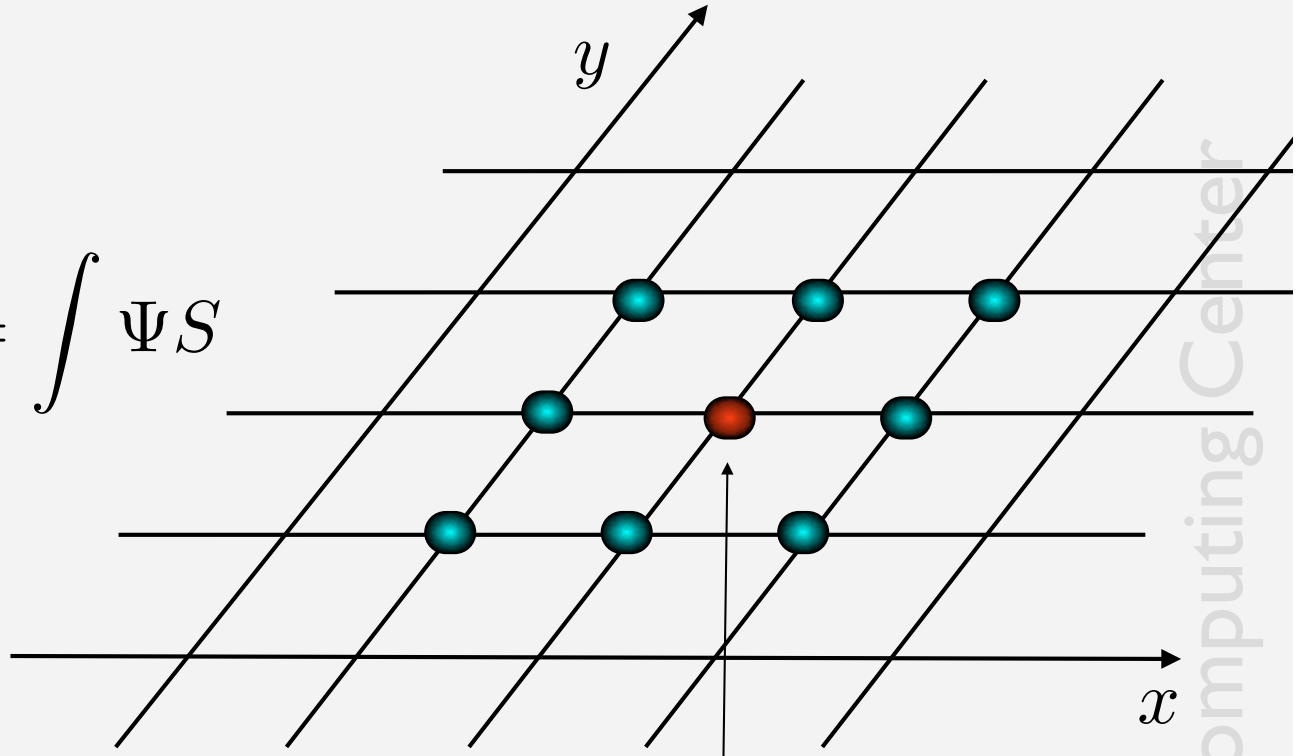
The fluxes on faces are function of the values of the unknown at both sides of the face.

How to increase the space-time order?

Barcelona Supercomputing Center

Third strategy:
Finite Elements

$$\int \Psi \frac{\partial \Phi^{\alpha}}{\partial t} + \int \Psi \frac{\partial F_i^{\alpha}}{\partial x_i} = \int \Psi S$$

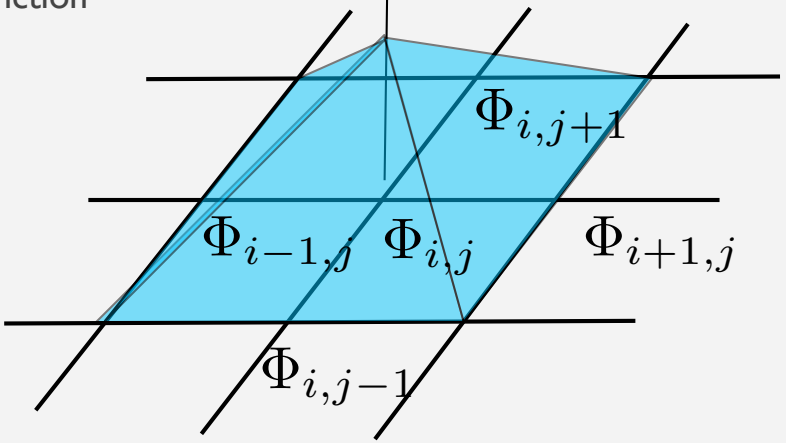


Divergence Gauss
Theorem

$$\int \Psi \frac{\partial F_i^{\alpha,1}}{\partial x_i} + \int \frac{\partial \Psi}{\partial x_i} F_i^{\alpha,2} - \oint \Psi F_i^{\alpha,2} n_i$$

Test Function

Ψ



Barcelona Supercomputing Center

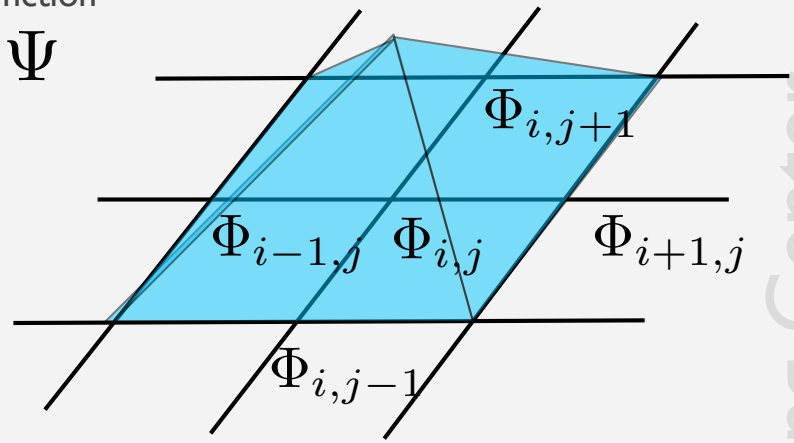
Discretizing the System

Third strategy:
Finite Elements

$$\int \Psi \frac{\partial F_i^{\alpha,1}}{\partial x_i} + \int \frac{\partial \Psi}{\partial x_i} F_i^{\alpha,2} - \oint \Psi F_i^{\alpha,2} n_i$$

Test Function

Ψ



As in FV, the test function is a "filter" that focuses the equation only in the filter's support.

When you project, the rest of the domain disappears.

Is like the IF, but filtered on small cells, the elements.

You only have to compute the fluxes through the limits of the cell or pass the derivatives (convolution) to the filter

The integrals are computed **numerically** as function of the values at the nodes of the elements

How to increase the space-time order?

Barcelona Supercomputing Center

Finite Differences

Very intuitive

Simple codification

Very efficient and well suited for translation to a computer code

Structured meshes

Problems with Neuman boundary conditions (but solvable, if you pay the price)

Lack of scalability for high order schemes

Humble numerics: stabilization, boundary conditions, adaptivity...

Barcelona Supercomputing Center

Finite Volumes

Intuitive, specially for Physicists

More complex codification

Very robust, with large story of success and theoretical developments

Unstructured meshes

Convective fluxes, ok. Diffusive fluxes, more bricolage is needed (but doable)

Lack of scalability for high order schemes

Sophisticate numerics: stabilization, boundary conditions, adaptivity...

Long tradition, particularly for CFD compressible flows

Barcelona Supercomputing Center

Finite Elements

Not very intuitive

Apparent complex codification, but not that much...

Profound mathematical background

Unstructured meshes, hybrid meshes

High order schemes naturally implemented

Good scalability

Not necessary expensive: must be programmed with care

Very sophisticate numerics: stabilization, boundary conditions, adaptivity...

Long tradition in all fields of PDEs

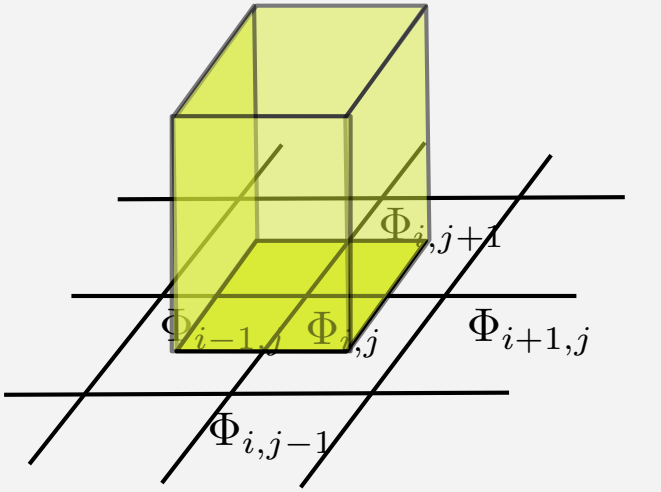
Barcelona Supercomputing Center

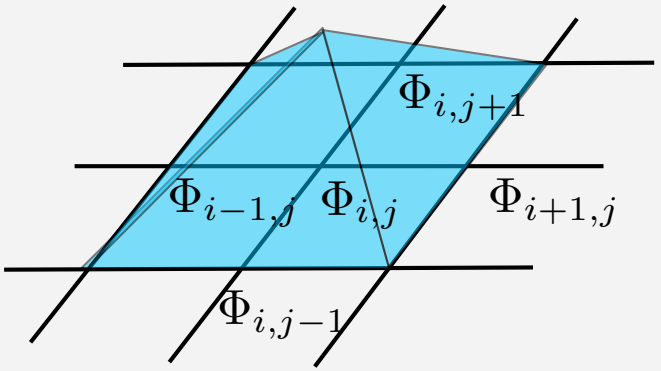
Unstructured meshes

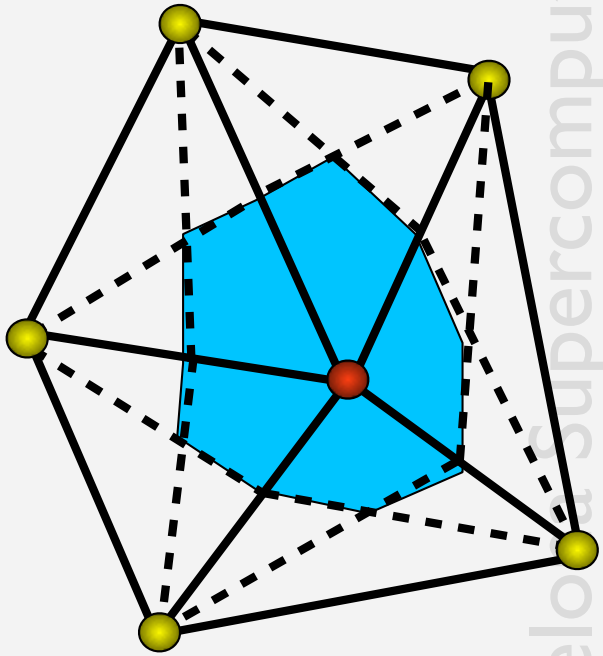
Definition

How to construct the characteristic function or the test function?

How to increase the order?







Barcelona Supercomputing Center

Unstructured meshes

Definition

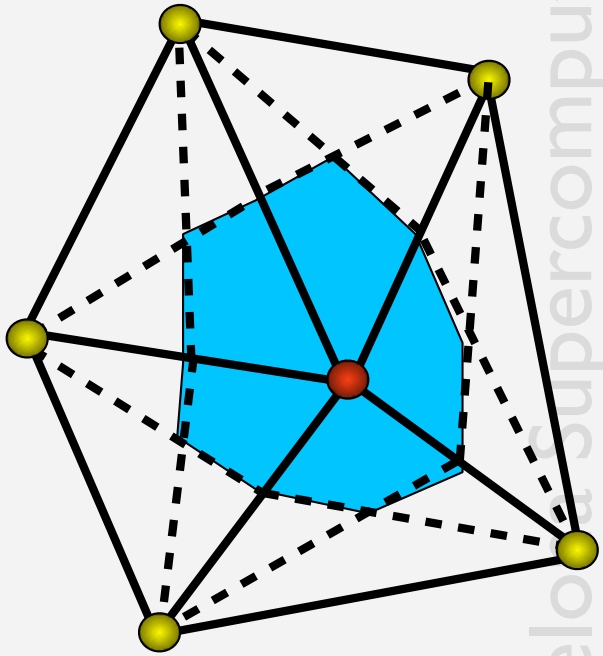
How to construct the characteristic function or the test function?

How to increase the order?

What are the advantages?

How you generate the mesh?

How do you implement?



Barcelona Supercomputing Center

Bibliography

Finite Differences:

Any basic book on numerical solutions of PDEs

Finite Volumes:
Look also at books on compressible flow CFD (Anderson, Courant, Chapman...)

A mathematical introduction to Fluid Mechanics. J. Chorin and J. Marsden. Springer. 1993
Finite volume methods for hyperbolic problems. R.J. Leveque. Cambridge. 2004

Finite Elements:
Mostly written by engineers...

The finite element method. O.C. Zienkiewicz and R. Taylor. Butterworth Heinemann. 2000
The finite element method. T.J.R. Hughes. Dover. 2000

Barcelona Supercomputing Center

Discretization:

Algorithms and Codes

Barcelona Supercomputing Center

Systems governed by PDEs coming from conservation principles

Forces, Energy, Mass... + Boundary and initial conditions

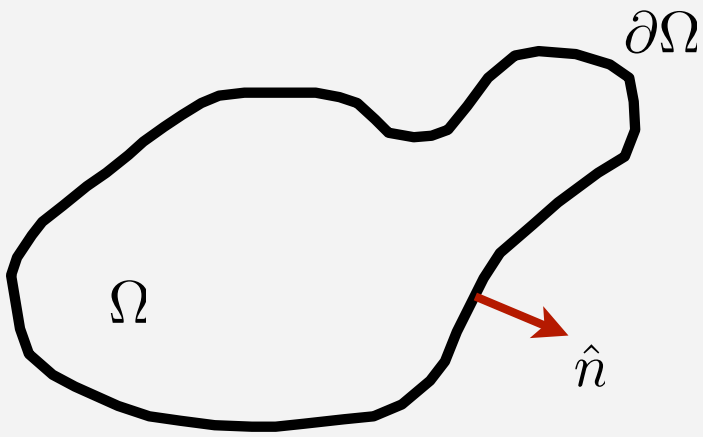
$$\frac{\partial q^{\alpha}}{\partial t} - \frac{\partial F_i^{\alpha}(q^{\beta})}{\partial x_i} = S$$

The Differential Conservative Form (DCF)

$q^{\alpha}(x_i, t)$ ← Fields of unknowns

$F_i^{\alpha}(q^{\beta}, x_i, t)$ ← Fluxes

$S(x_i, t)$ ← Sources



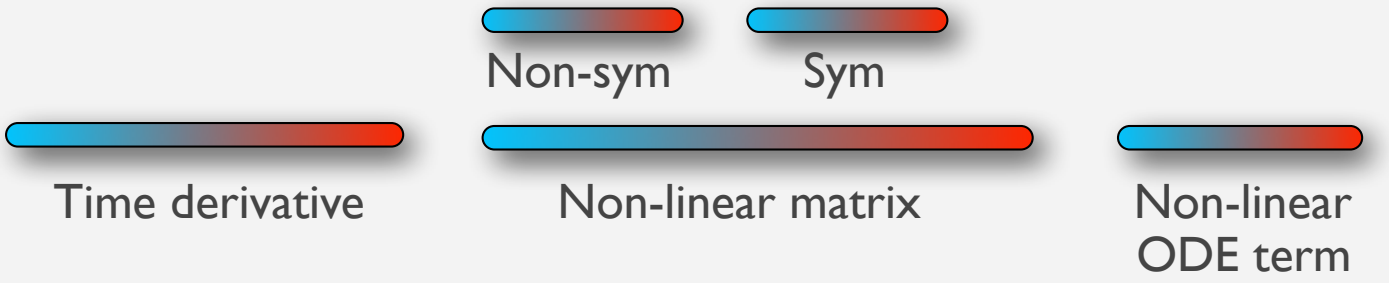
Barcelona Supercomputing Center

Discretizing the System

Discretization issues:

We arrive at the following matrix system

$$M\frac{\Phi^{n+1}-\Phi^{n}}{\Delta t}+\left[C(\Phi)+K(\Phi)\right]\Phi^{*}+s(\Phi^{*})=b$$



We must adopt a solution strategy (the **solver**), analyzing these issues

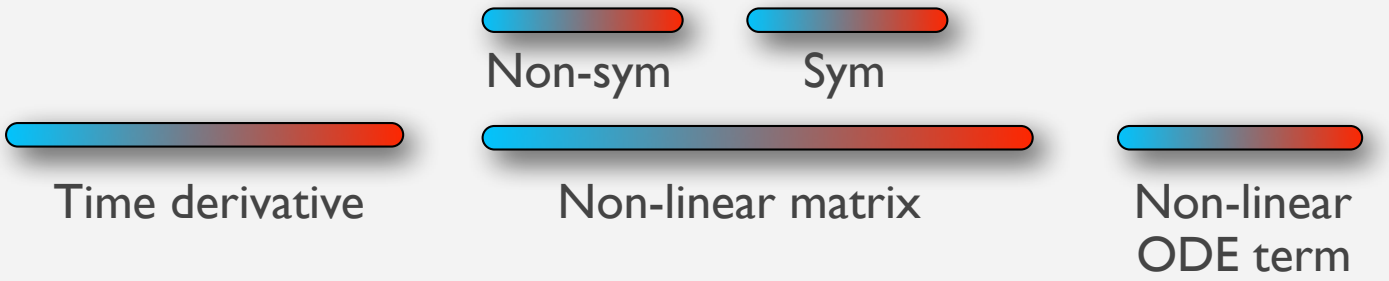
- ☑ Explicit
- ☑ Implicit
- ☑ Solving "by blocks"
- ☑ How and what to parallelize

Barcelona Supercomputing Center

Discretization issues:

We arrive at the following matrix system

$$M\frac{\Phi^{n+1}-\Phi^{n}}{\Delta t}+\left[C(\Phi)+K(\Phi)\right]\Phi^{*}+s(\Phi^{*})=b$$



Depending on the definition of * the scheme is **Explicit** or **Implicit**

$$*:=n+1$$ ← Implicit

$$*:=n$$ ← Explicit

Barcelona Supercomputing Center

Explicit schemes

$$M_d \frac{\Phi^{n+1} - \Phi^n}{\Delta t} = -\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b$$

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t\left(\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b\right)$$

Features:

RHS can be directly computed, no matrix storage required

Computing the RHS (assembly) is, by far, the most time consuming part

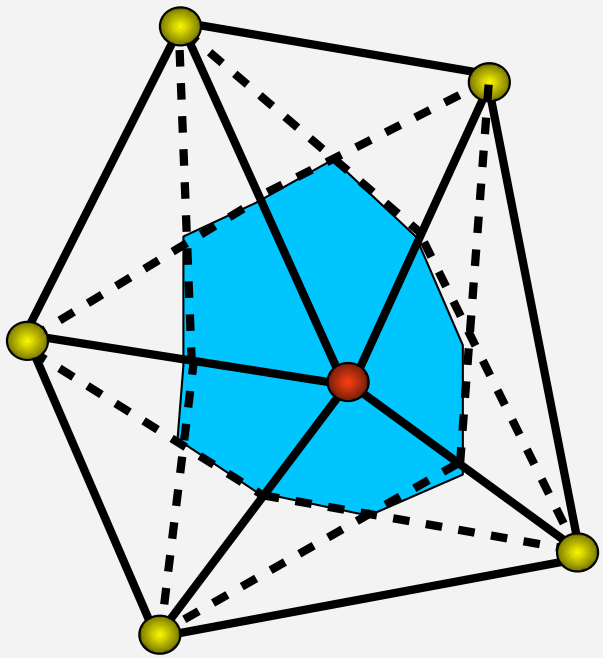
Barcelona Supercomputing Center

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t\left(\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b\right)$$

Features:

Lumped mass matrix represents the "mass" associated to each mesh node

It is a diagonal matrix, trivially inverted



Barcelona Supercomputing Center

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

What is the time a signal takes to propagate within a given element?

- ☑ Velocity
- ☑ Diffusion
- ☑ Acoustic waves (linear, small perturbations)
- ☑ Shock waves (non linear, strong gradients)

For instance

$$\Delta t = \frac{f^{\mathrm{CFL}}}{\frac{u}{h_1} + \frac{2k}{h_2^2} + \frac{c}{h_3}}$$

Barcelona Supercomputing Center

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t\left(\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b\right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

From theoretical arguments for simple equations, the CFL factor is 1.

However, more complex problems could require smaller figures.

But what is "h"?

$$\Delta t = \frac{f^{\mathrm{CFL}}}{\frac{u}{h_1} + \frac{2k}{h_2^2} + \frac{c}{h_3}}$$

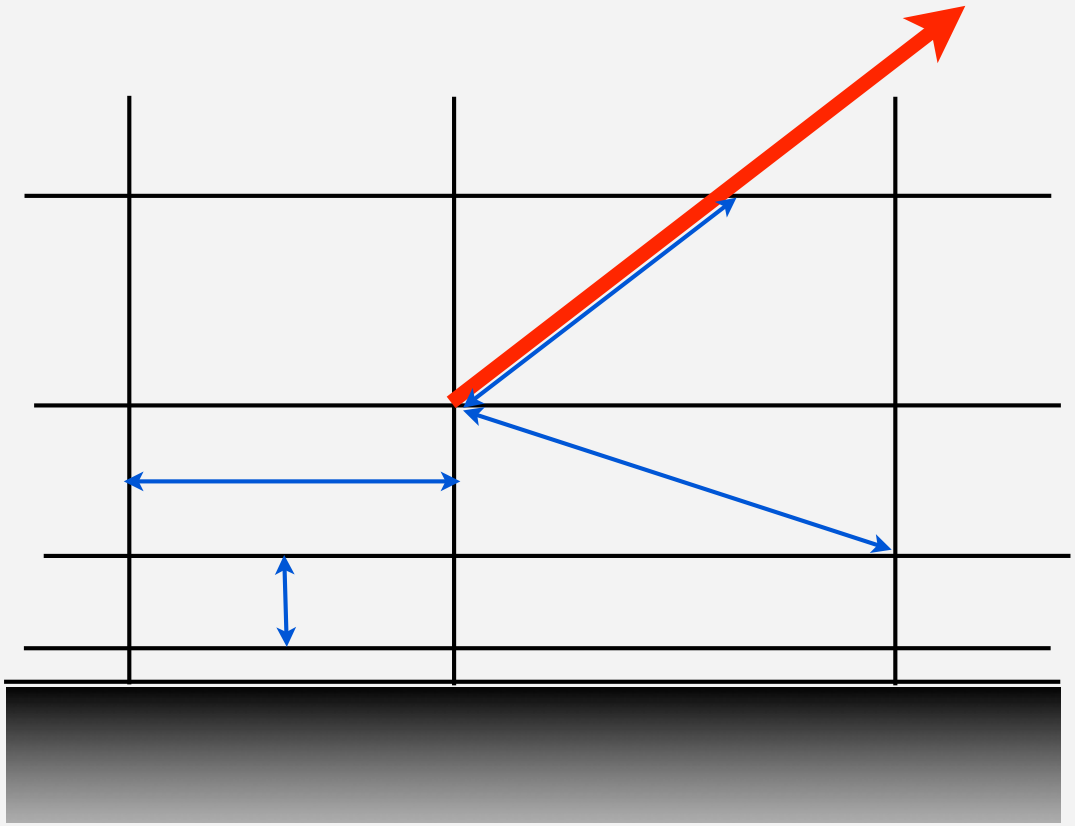
Barcelona Supercomputing Center

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

But what is "h"?



Barcelona Supercomputing Center

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

For transient problems, the mesh minimum value is taken: it can be VERY SMALL!!!

For transient coupled problems, the per-equation minimum value is taken

For stationary problems, local time steps can be used

Both can be combined with "pseudo-time step" formulations (Jameson papers)

Barcelona Supercomputing Center

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t\left(\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b\right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

The Characteristic Condition Number is the ratio of the largest to the smallest characteristic speeds:

$$u, \frac{2k}{h}, c, \ldots$$

When one of them goes to zero, the system becomes extremely "stiff"

Preconditioning is required!

☑ Local preconditioners for Explicit schemes: Turkel, Weiss, Van Leer, ...

☑ Global preconditioners for Implicit iterative schemes: Diagonal, ILU, ...

Barcelona Supercomputing Center

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Dirichlet boundary conditions can be imposed after a time advance step

This gives enough flexibility for non-linear conditions: Navier-Stokes / Euler

Neuman boundary conditions enters in the RHS for FEM

Neuman boundary conditions are transformed in Dirichlet ones for FD

Barcelona Supercomputing Center

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

But some problems are inherently stiff... such as incompressible flows or solids

Worse: coupling multi-physics with different time scales

Then... Implicit methods are the best option.

Barcelona Supercomputing Center

$$M\frac{\Phi^{n+1}-\Phi^{n}}{\Delta t}+\left[C(\Phi)+K(\Phi)\right]\Phi^{*}+s(\Phi^{n+1})=b$$

$$\left[M\frac{1}{\Delta t}+M_d\,s(\Phi)+C(\Phi)+K(\Phi)\right]\Phi^{n+1}=M\frac{\Phi^{n}}{\Delta t}+b$$

Time ODE-like Non-sym Sym RHS

Non-linear matrix

Barcelona Supercomputing Center

$$\left[M \frac{1}{\Delta t} + M_d \, s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M \frac{\Phi^n}{\Delta t} + b$$

Sub-matrices:

Time matrix: consistent mass matrix, that couples first neighbors (for first order)

ODE-like matrix: diagonal mass matrix to preserve locality. Let us suppose this term non-linear

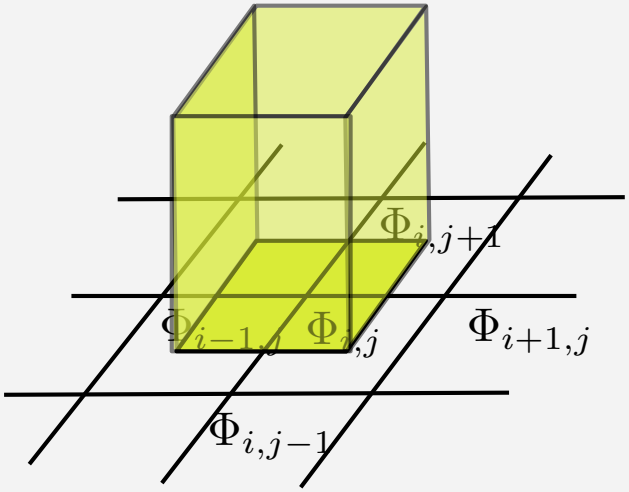
Non-symmetric matrix: comes from first space derivative matrices, such as convection ones

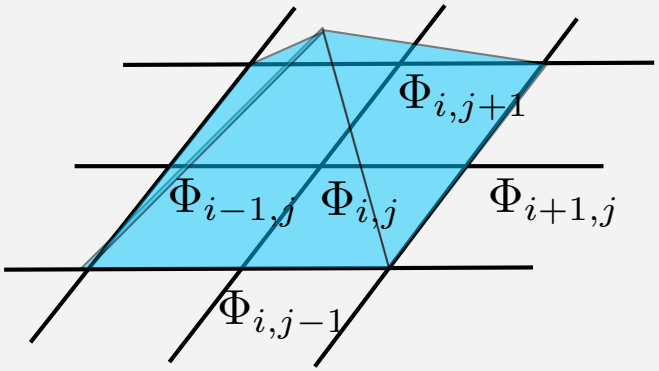
Symmetric matrix: comes from second space derivative matrices, such as Laplacians (diffusion) or stress (large strain solid mechanics)

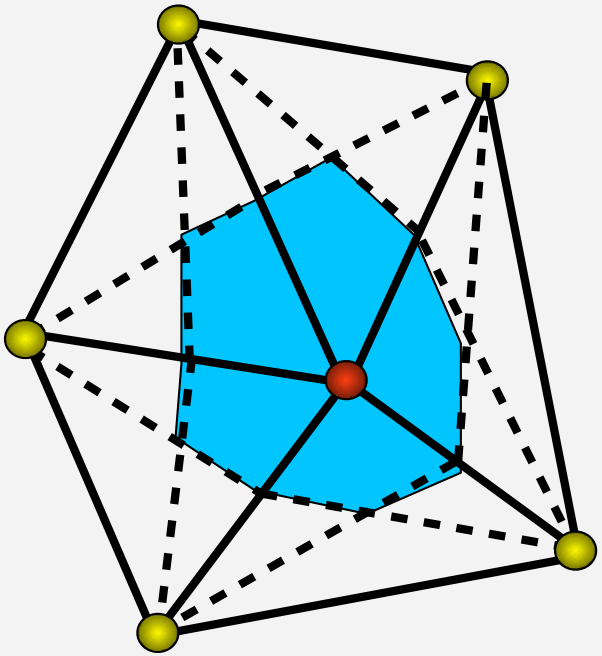
Barcelona Supercomputing Center

$$\left[M \frac{1}{\Delta t} + M_d \; s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M \frac{\Phi^n}{\Delta t} + b$$

What is the structure of these matrices?







Barcelona Supercomputing Center

$$\left[M \frac{1}{\Delta t} + M_d \, s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M \frac{\Phi^n}{\Delta t} + b$$

Features:

Consider the matrix divided in blocks according to certain grouping of the unknowns. Then, not all the blocks prefer the same solution scheme...

Typically, compact support problems can have a relatively low bandwidth, so sparse algebra is better-suited.

On the other hand, other problems can produce filled matrices (i.e. Fourier, Bessels, plane waves...)

Renumbering can be decisive for a better data distribution

If conservative forms are required due to the presence of discontinuities, the implicit form can be tricky

Barcelona Supercomputing Center

$$\left[M \frac{1}{\Delta t} + M_d \, s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M \frac{\Phi^n}{\Delta t} + b$$

Features:

The problem is now clearly divided in two parts

1. The RHS and matrices assembly
2. The solver

Which is the most time consuming part in an implicit scheme?

Typically the second one, say from 40-60 up to 10-90

Let us analyze the structure of a simulation code…

Barcelona Supercomputing Center

Solution strategies

Input data

Compute geometrical stuff (only once)

Do time steps

  Do coupling iterations

    Do linear iterations

      Compute Matrix and RHS:

      Do elements (or faces or nodes...) iterations

        Compute elementary Matrix and RHS

        Assemble

      Enddo

      Solver (iterative or direct)

    Enddo

    Update coupling

    Enddo

  Enddo

  Update time step

Enddo

Output data

$$\left[M \frac{1}{\Delta t} + M_d \, s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M \frac{\Phi^n}{\Delta t} + b$$

Barcelona Supercomputing Center

Solution strategies

Input data

Compute geometrical stuff (only once)

Do time steps

  Do coupling iterations

    Do linear iterations

      Compute Matrix and RHS:

      Do elements (or faces or nodes...) iterations

        Compute elementary Matrix and RHS

        Assemble

      Enddo

      Solver (iterative or direct)

    Enddo

    Update coupling

    Enddo

  Enddo

  Update time step

Enddo

Output data

Mesh
Boundary conditions
Scalar data

Binary formats for large data
(Endian vs Big Endian)

Standard formats such as CGNS or
marked for scalar

Checkpoint - restart

Barcelona Supercomputing Center

Solution strategies

Input data

Compute geometrical stuff (only once)

Do time steps

  Do coupling iterations

    Do linear iterations

      Compute Matrix and RHS:

      Do elements (or faces or nodes...) iterations

        Compute elementary Matrix and RHS

        Assemble

      Enddo

      Solver (iterative or direct)

    Enddo

    Update coupling

    Enddo

  Enddo

  Update time step

Enddo

Output data

Stencils and mass matrix

Analyze matrix system to see what can be pre-computed

Trade off between storage and computing time

Compute normals and boundary values

Barcelona Supercomputing Center

Solution strategies

Input data

Compute geometrical stuff (only once)

Do time steps

Do coupling iterations

Do linear iterations

Compute Matrix and RHS:

Do elements (or faces or nodes...) iterations

Compute elementary Matrix and RHS

Assemble

Enddo

Solver (iterative or direct)

Enddo

Update coupling

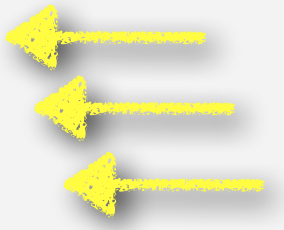
Enddo

Enddo

Update time step

Enddo

Output data



$$\left[M\frac{1}{\Delta t} + M_d\, s(\Phi) + C(\Phi) + K(\Phi)\right]\Phi^{n+1} = M\frac{\Phi^n}{\Delta t} + b$$

Barcelona Supercomputing Center

Solution strategies

Input data

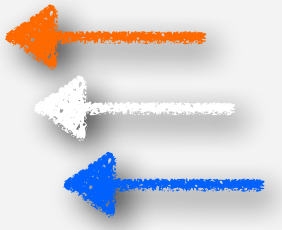
Compute geometrical stuff (only once)

Do time steps

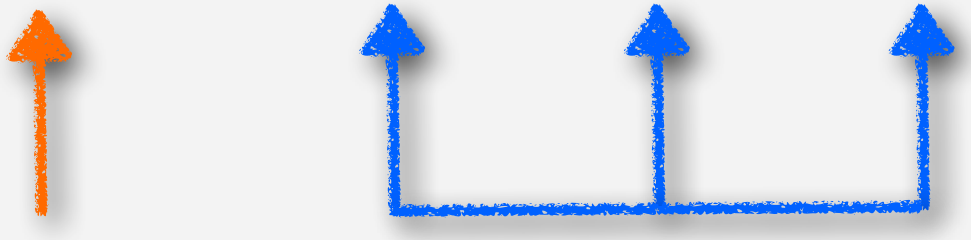
Do coupling iterations

Do linear iterations

Compute Matrix and RHS



$$\left[M \frac{1}{\Delta t} + M_d \, s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M \frac{\Phi^n}{\Delta t} + b$$



Coupling:
- related to the solver
- related to the Physics
- related to both

Barcelona Supercomputing Center

Solution strategies

Input data

Compute geometrical stuff (only once)

Do time steps

Do coupling iterations

Do linear iterations

Compute Matrix and RHS:

Do elements (or faces or nodes...) iterations

Compute elementary Matrix and RHS

Assemble

Enddo

Solver (iterative or direct)

Enddo

Update coupling

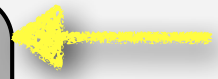
Enddo

Enddo

Update time step

Enddo

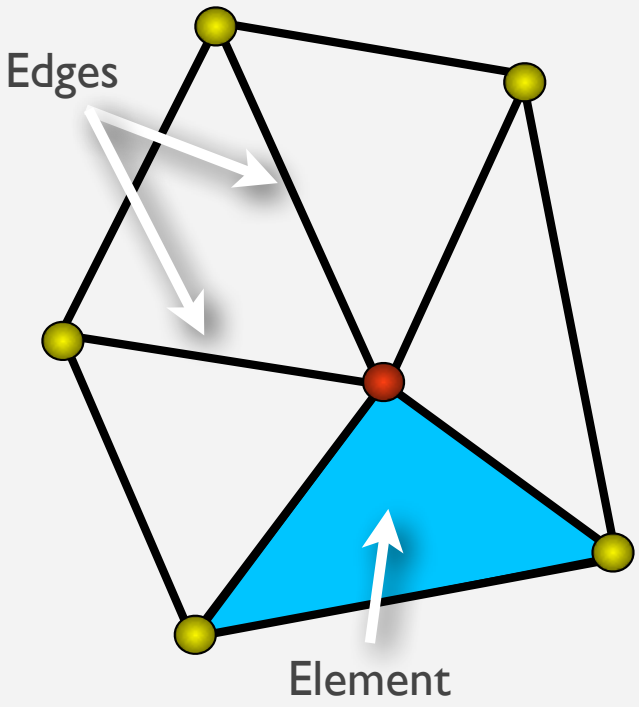
Output data

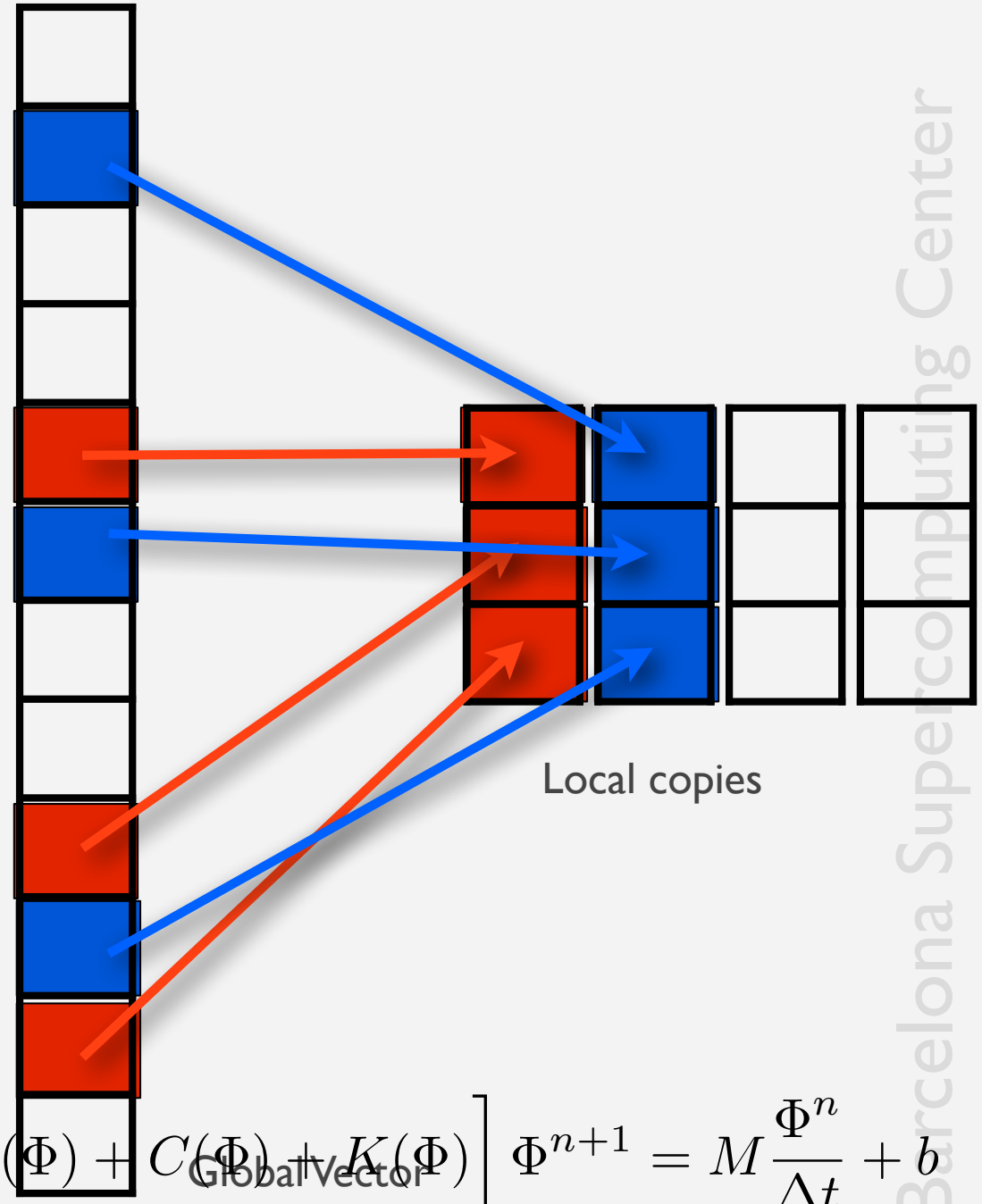


Loop over element, nodes, faces, edges, to compute local contributions to RHS and Matrices

Locality of data (gather - scatter)

Barcelona Supercomputing Center





Make a local copy of the working vectors
The more FLOPs per DOF, the better is to do a local copy
Local copies can be done every N elements

$$\left[M \frac{1}{\Delta t} + M_d \, s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M \frac{\Phi^n}{\Delta t} + b$$

Barcelona Supercomputing Center

Solution strategies

Input data

Compute geometrical stuff (only once)

Do time steps

  Do coupling iterations

    Do linear iterations

      Compute Matrix and RHS:

      Do elements (or faces or nodes...) iterations

        Compute elementary Matrix and RHS

        Assemble

      Enddo

      Solver (iterative or direct)

    Enddo

    Update coupling

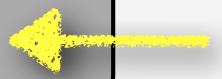
    Enddo

  Enddo

  Update time step

Enddo

Output data



For implicit schemes, this is the bottle neck

Newton - Krylov methods
Jacobi iterations
etc.

Barcelona Supercomputing Center

Solvers

Direct solvers or iterative solvers?

Avoid direct solvers when possible
- Good for small - medium problems
- Bad for large ones, too much memory required
- Very bad scalability
- However, very stiff problems could require a direct solver...

The election of the iterative solver is strongly biased by the form of the matrix:

Symmetric, then Conjugate Gradient family
Otherwise, OrthoMin, GMRES, BiCGStab…

Barcelona Supercomputing Center

Solvers

Iterative solvers

☑ *Iterative methods for sparse linear systems*. Y. Saad. 2000
☑ *Newton-Krylov methods: a survey of approaches and applications*. D. Knoll and D. Keyes. JCP. 2004

Sparse, large matrices
Involves matrix-vector and vector-vector products

Stand-alone or matrix-free versions

Issues:
- Linearization
- Boundary conditions
- Preconditioners
- Solution strategy

Barcelona Supercomputing Center

Issues: Linearization

From **Jacobi** (well suited for fluids) up to **Newton methods**, with analytic or approximated tangent matrix (required in solid mechanics)

See that in solid mechanics, the total Lagrangian formulation for large deformation problems with dynamic terms is

$$\rho_o \frac{\partial^2 u_i}{\partial t^2} = \frac{\partial P_{iJ}}{\partial X_J} + \rho_o B_i$$

$$\int_{\Omega_o} \Phi \rho_o \frac{\partial^2 u_i}{\partial t^2} = -\int_{\Omega_o} \frac{\partial \Phi}{\partial x_J} P_{iJ} + \int_{\partial \Omega_o} N_J P_{iJ} \Phi + \int_{\Omega_o} \Phi \rho_o B_i$$

Barcelona Supercomputing Center

Issues: Linearization

$$\int_{\Omega_o} \Phi \rho_o \frac{\partial^2 u_i}{\partial t^2} = -\int_{\Omega_o} \frac{\partial \Phi}{\partial x_J} P_{iJ} + \int_{\partial \Omega_o} N_J P_{iJ} \Phi + \int_{\Omega_o} \Phi \rho_o B_i$$

$$F_{iJ} = \frac{\partial x_i}{\partial X_J} = \delta_{iJ} + \frac{\partial u_i}{\partial X_J}$$

Deformation gradient tensor

$$P_{iJ} = F_{iK} S_{KJ}$$

1st. Piola Kirchoff

$$S_{KJ}(F)$$

2nd. Piola Kirchoff:
Very complex dependence on the deformation gradient!!!!

Barcelona Supercomputing Center

Issues: Boundary conditions

$$\left[M\frac{1}{\Delta t} + M_d\, s(\Phi) + C(\Phi) + K(\Phi)\right]\Phi^{n+1} = M\frac{\Phi^n}{\Delta t} + b$$

Some times Dirichlet boundary conditions must be imposed indirectly, on derived variables

For instance, Compressible Navier-Stokes are solved on conservative variables, while BCs are imposed in primitive variables

Then, you will need to change variables locally (using either a Jacobian or a rotation matrix.. or both)

Barcelona Supercomputing Center

Issues: Preconditioners

"A preconditioner is any form of implicit or explicit modification of an original linear system which makes it "easier" to solve by a given iterative method." (from Saad)

$$Ax = b$$

$$M^{-1} Ax = M^{-1} b$$

Example: local preconditioners in low-Mach compressible flow

To take into account:
Trade off between the cost of computing the preconditioner and how good it is to improve robustness and efficiency

Particularly important for parallelized codes!

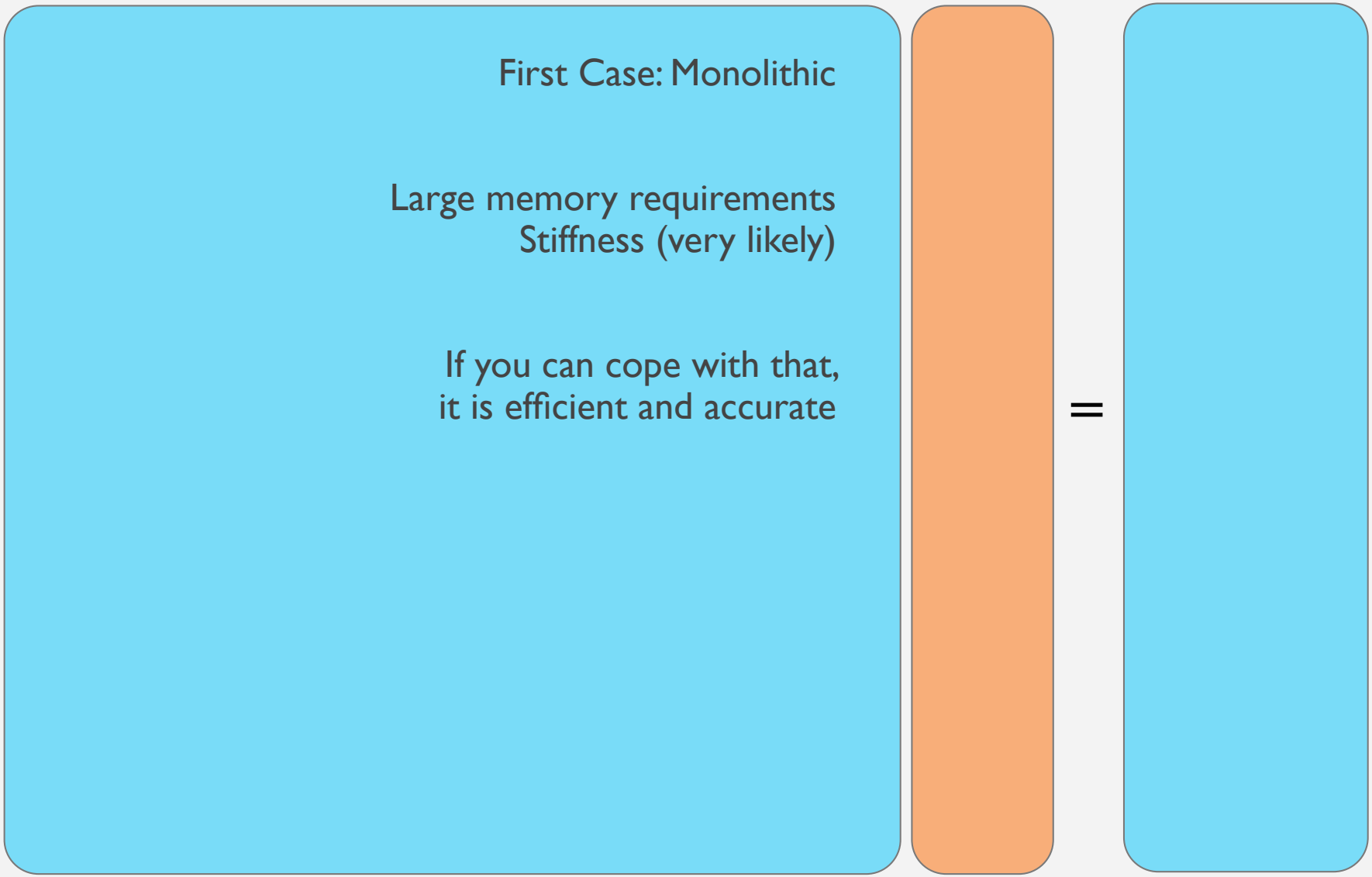
Barcelona Supercomputing Center

Issues: Solution strategy

$$\left[M\frac{1}{\Delta t} + M_d\, s(\Phi) + C(\Phi) + K(\Phi)\right]\Phi^{n+1} = M\frac{\Phi^n}{\Delta t} + b$$

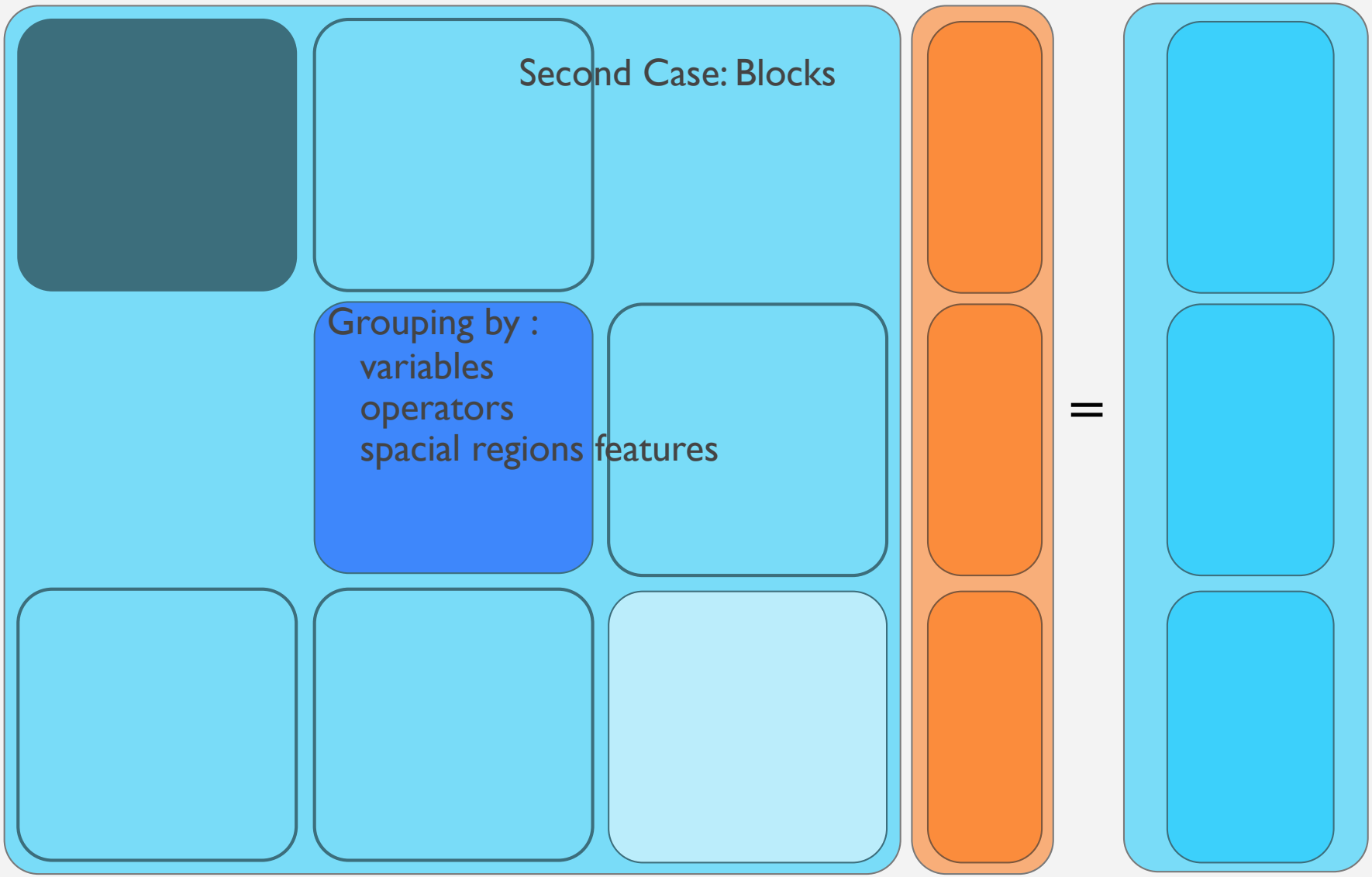
Barcelona Supercomputing Center

Issues: Solution strategy



Barcelona Supercomputing Center

Issues: Solution strategy



Barcelona Supercomputing Center

Post-process

Input data

Compute geometrical stuff (only once)

Do time steps

  Do coupling iterations

    Do linear iterations

      Compute Matrix and RHS:

      Do elements (or faces or nodes...) iterations

        Compute elementary Matrix and RHS

        Assemble

      Enddo

      Solver (iterative or direct)

    Enddo

    Update coupling

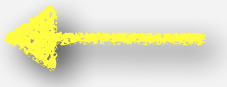
    Enddo

  Enddo

  Update time step

Enddo

Output data



Barcelona Supercomputing Center

Post-process

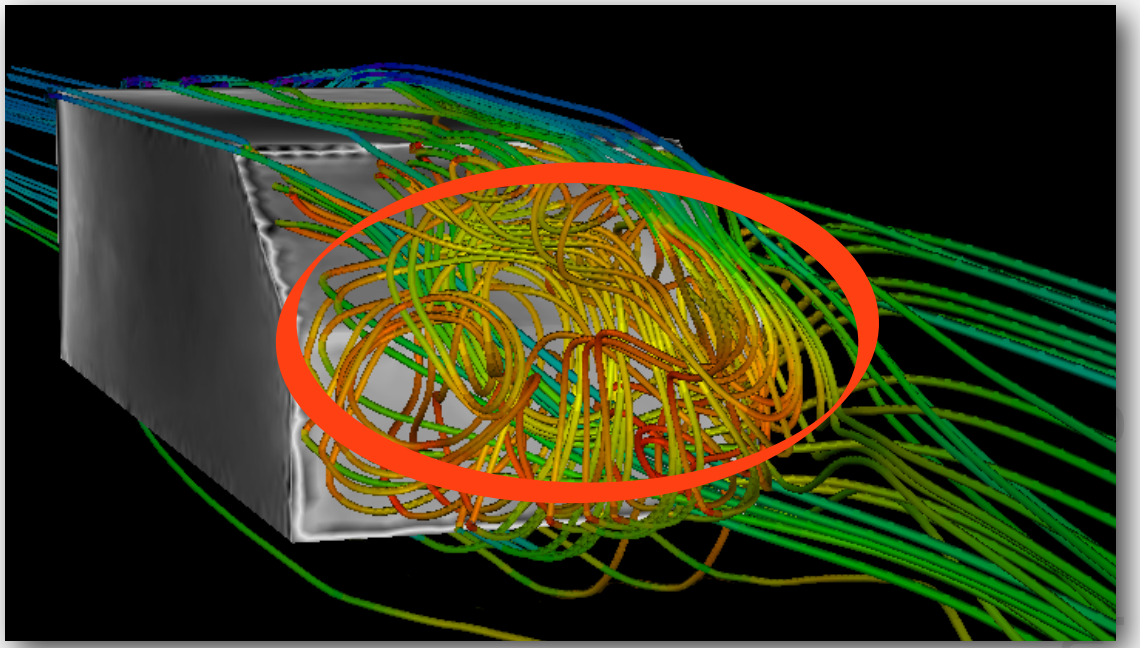
Output data

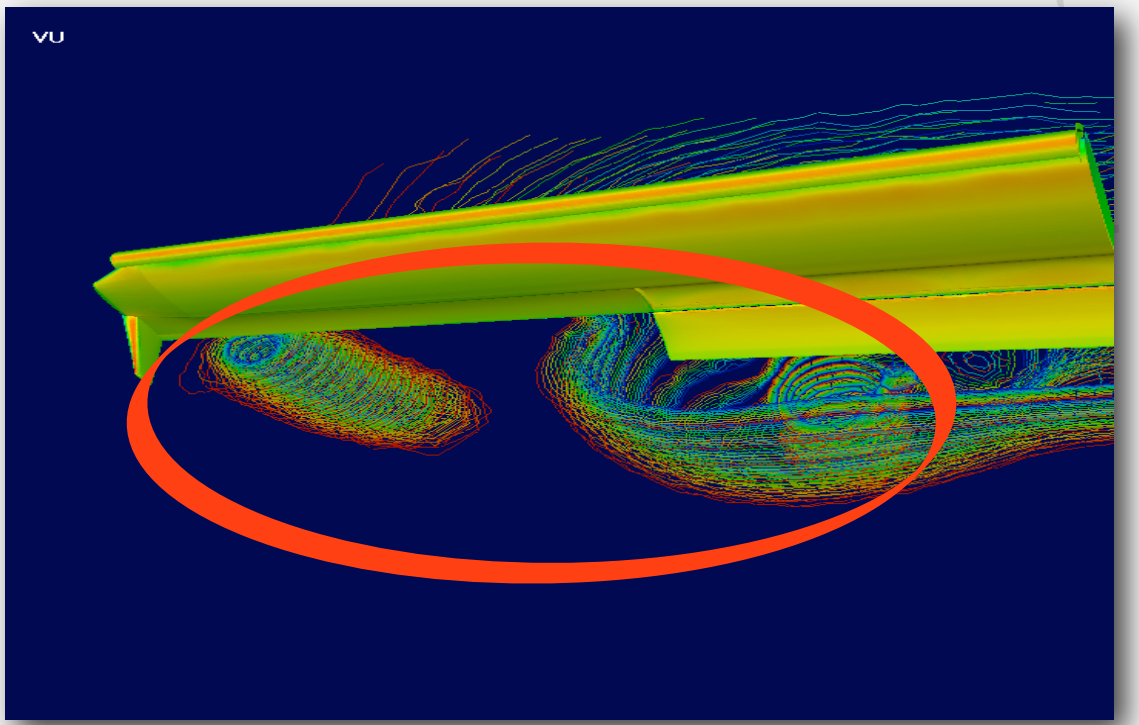
Visualizers

Widely used formats
Filtering and smart data extraction
Checkpoint - restart

Parallel I/O
HDF5

Hiper-realistic
Use of "video games" techniques





Output data

Visualizers

Widely used formats
Filtering and smart data extraction
Checkpoint - restart

Parallel I/O
HDF5

Hiper-realistic
Use of “video games” techniques

Barcelona Supercomputing Center